University of Victoria
CSC 360: Operating Systems
Dr. Jianping Pan

ASSIGNMENT P2A
MULTI-THREADED SCHEDULER DESIGN DOCUMENT

Due: March 7th, 2018

William H. Grosset
V00820930

1.
N + 1 threads will be used, where N represents how many trains will be in the input file and the + 1 being the main thread. Each train thread will be in charge of its own loading and crossing time and the main thread is in charge of dispatching trains (at most 1) across the track.

2.
Trains load concurrently via threads, but the main thread acts as the controller thread. The main thread decides when all trains will be initially loaded and when a train will cross the track.

3.
3 mutexes:

- Load mutex (guards access to begin loading)
- West station mutex (guards access to the west station priority queue)
- East station mutex (guards access to the west station priority queue)

4.
The main thread is either dispatching trains or waiting for a signal from a station (idle).

5.
Priority (FIFO) queue for **each** station to represent the trains that are ready to cross.

6.
A mutex to guard multiple threads from being able to enqueue/dequeue from a station priority queue.

7.
N + 3 condition variables, where N is the number of trains/threads (each have `can_cross`), and `can_load`, `threads_created`, `station_ready` are the 3 global condition variables.

Convar: `can_load`

a) The condition represents if a train can begin loading.
b) Track mutex — gateway to begin loading
c) After `pthread_cond_wait()`: Each thread will begin to `usleep` for their appropriate loading time and then are added to the station priority queue.

Convar: `threads_created`

d) Condition represents if the last train has been created.
e) Track mutex — dispatcher waits to broadcast loading

f) After `pthread_cond_wait()`: Last train signals dispatcher and allows the main thread to broadcast to each thread to begin loading.

Convar: `station_ready`

g) The condition represents if a train is ready in the station queue.
h) Track mutex — dispatcher waits to be signaled by a ready train
i) After `pthread_cond_wait()`: The main thread will `peek()` on both queues and decide which train to release (based on the simulation rules) to cross the track.

Convar: `can_cross`

j) The condition represents if a train can cross.
k) Track mutex — only one thread can access the track
l) After `pthread_cond_wait()`: The targeted thread will begin to `usleep` for their appropriate crossing time and then signals the main thread once finished.

8.

Main Thread = MT; Train Thread = TT

- MT: Lock track mutex, read input file, create threads for each train
- TT: If last train, signal main thread, otherwise lock track mutex and wait to load
- MT: Waits for signal from last thread → broadcast to all trains to begin loading
- TT: Begins loading and locks station mutex, enqueues, releases station mutex
- MT: Check if PQs empty and/or wait for signal to dispatch
- TT: Signals to dispatcher that train is ready, waits for signal to cross
- MT: Receives signal, check both PQs and signals/dispatches appropriate train
- TT: Lock track mutex, begins crossing, signals dispatcher and releases mutex, exit
- MT: Waits for signal and repeats step 5 until all threads have finished, exit
- MT: Repeat Step 5 until all threads have finished → exit